Using Software Metrics as Demonstrators of Design Changes in Iterative Software Processes

Mohammad Alshayeb
Computer Science Department
University of Alabama in Huntsville
April 9, 2002

Agenda
- Research Objectives
- Research Hypotheses
- Metrics Suites
- Previous Software Metrics Studies
- Software Processes
- Research Data
- Research Approach
- Research Significance

Research Objectives
How object-oriented (OO) software metrics can be used in iterative software processes to:
- Predict source line changes within each class from one iteration to the next.
- Reveal how OO design structures evolve in iterative processes and compare the evolution patterns with previous studies.
- Validate software metrics including the design instability metric.

Research Hypotheses
Hypothesis 1: Using OO metrics, we can predict source line changes in classes from one iteration to the next in the long-cycled framework iterative process.
Hypothesis 2: Using OO metrics, we can predict source line changes in classes from one iteration to the next in the short-cycled Extreme Programming (XP) iterative process.
Hypothesis 3: Using OO metrics, we can predict maintenance effort, measured in man-hour, in classes from one iteration to the next in the short-cycled XP iterative process.
Hypothesis 4: Using OO metrics, we can predict refactoring effort, measured in man-hour, in classes from one iteration to the next in the short-cycled XP iterative process.
Hypothesis 5: The design evolution should exhibit similar patterns in the short- and long-cycled iterative processes as it did in previous non-iterative studies.
Hypothesis 6: System Design Instability (SDI) can indicate project progress in both the short- and the long-cycled iterative process as it did in the previous study for the non-iterative process.
Research Hypotheses

Hypothesis 7: Class size has a strong impact on predicting design changes in the long-cycled framework iterative process.

Hypothesis 8: Class size has a strong impact on predicting design changes in the short-cycled XP iterative process.

Introduction

Metrics
Software metrics
Object-oriented software metrics
Software processes

Metrics Suites

Chidamber and Kemerer
- WMC, DIT, NOC, CBO, RFC and LCOM

Li
- NAC, NDC, NLM, CMC, CTA and CTM

Previous Software Metrics Studies

Li and Henry (J. of systems and software 1993).
Basili et al. (TSE 1996).
Briand et al. (TSE 1999).
Fenton and Neil (TSE 1999).
Fenton and Ohlsson (TSE 2000).
El Emam et al. (TSE 2001)
Briand and Wüst (TSE 2001)

Software Processes

Framework Iterative Process

Framework system:
- Collection of classes.
- Built into a cohesive inheritance hierarchy,
- Reusable and semi-complete application,
- Provide more comprehensive reuse than classes developed by individual programmers.
- Shipped as components used to build applications.
- Examples: CORBA, MFC and JFC.

Kent Beck 1999
Extreme Programming (XP)
- A new software process.
- Convenient and effective for projects that have vague requirements or the requirements are likely to change during development.
- XP features: stories, pair programming, unit testing and continuous integration.
- XP activities: New design, Error fix and Refactoring

Research Data
To answer hypotheses 1 and 7:
- Hypothesis 1: Using OO metrics, we can predict source line changes in classes from one iteration to the next in the long-cycled framework iterative process.
- Hypothesis 7: Class size has a strong impact on predicting design changes in the long-cycled framework iterative process.
- Various releases of JDK (JDK 1.0, JDK 1.1, JDK 1.2, JDK 1.3 and JDK 1.4)
  - Long evolutionary history,
  - Widely used in industry,
  - JDK releases changed throughout their development process,
  - Open source,

Research Data
To answer hypotheses 2, 3, 4, and 8:
- Hypothesis 2: Using OO metrics, we can predict source line changes in classes from one iteration to the next in the short-cycled (XP) iterative process.
- Hypothesis 3: Using OO metrics, we can predict maintenance effort, measured in man-hour, in classes from one iteration to the next in the short-cycled XP iterative process.
- Hypothesis 4: Using OO metrics, we can predict refactoring effort, measured in man-hour, in classes from one iteration to the next in the short-cycled XP iterative process.
- Hypothesis 8: Class size has a strong impact on predicting design changes in the short-cycled XP iterative process.

Two OO systems:
- Developed using Java, 2-year period of time, Data collected in daily basis.

Research Data
To answer hypotheses 5 and 6
- Hypothesis 5: The design evolution should exhibit similar patterns in the short- and long-cycled iterative processes as it did in previous non-iterative studies.
- Hypothesis 6: System Design Instability (SDI) metric can indicate project progress in both the short- and the long-cycled iterative process as they did in the previous study for the non-iterative process.

Combination of the data we plan to use to answer the other hypotheses.

Research Data
Iterative Software Process
- Extreme Programming (XP)
- Framework Iterative

Data: Two Java Applications
Data: JDK Releases

Research Approach
- Regression Trees:
  - Data mining technique.
  - Builds partition tree of the data set.
- Multiple Linear Regression (MLR):
  $$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \epsilon$$
Regression Trees

Research Significance
- Empirical validation of metrics is very important if the metrics are to be used.
- Validation of the metrics in OO iterative processes has never been done before
  - XP: short iterative cycle
  - Framework: long iterative cycle
- Try different prediction models
  - Multiple Linear Regression (MLR)
  - Regression Trees

Research Significance
- Examine the effect of class size on the prediction models.
- Validate the system design instability (SDI) metric in the two iterative processes.
- Reveal how OO design structures evolve in the two iterative processes and compare the evolution with previous results.

Summary
- Three objectives.
- Eight hypotheses.
- Approach to test the hypotheses.

Prediction Example

| Class Name | Lines Deleted | CTM | NLM | WMC | CTA | LOC
|------------|---------------|-----|-----|-----|-----|-----
| Container  | 17            | 25  | 19  | 92  | 6   | 217 |
| Hashable   | 23            | 12  | 19  | 53  | 1   | 113 |
| InetAddress | 23            | 14  | 10  | 24  | 2   | 136 |
| Point      | 3             | 2   | 10  | 0   | 0   | 0   |
| Runtime    | 31            | 10  | 17  | 22  | 8   | 253 |
| String     | 18            | 22  | 18  | 126 | 8   | 373 |
| Thread      | 30            | 43  | 21  | 56  | 2   | 786 |
| Boolean    | 1             | 1   | 1   | 2   | 1   | 4   |

Given the following data: number of lines deleted from JDK1.1–JDK1.2 in each class.

Questions?
Prediction Example

The dependent variable:
- Lines deleted.

The independent variables are:
- CTM
- NLM
- WMC
- CTA
- LCOM

Determine which independent variables are significant predictors of the dependent variable, and which independent variables can be eliminated.
- Find the best subset.

Best Subsets Regression: Lines Changed versus CTM, NLM, WMC, CTA, LCOM

<table>
<thead>
<tr>
<th>R-sq</th>
<th>R-sq(adj)</th>
<th>C-p</th>
<th>S</th>
<th>MCAW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.3</td>
<td>5.2</td>
<td>10.79</td>
<td>X</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>5.7</td>
<td>11.009</td>
<td>X</td>
</tr>
<tr>
<td>2.2</td>
<td>0.0</td>
<td>6.0</td>
<td>9.9504</td>
<td>X</td>
</tr>
<tr>
<td>2.2</td>
<td>0.0</td>
<td>6.4</td>
<td>11.075</td>
<td>X X</td>
</tr>
<tr>
<td>3.0</td>
<td>0.0</td>
<td>6.9</td>
<td>9.0652</td>
<td>X X</td>
</tr>
<tr>
<td>3.0</td>
<td>0.0</td>
<td>5.3</td>
<td>9.0097</td>
<td>X X</td>
</tr>
<tr>
<td>4.0</td>
<td>0.0</td>
<td>4.2</td>
<td>11.028</td>
<td>X X X</td>
</tr>
<tr>
<td>4.0</td>
<td>0.0</td>
<td>4.7</td>
<td>9.2330</td>
<td>X X X</td>
</tr>
<tr>
<td>5.0</td>
<td>0.0</td>
<td>4.0</td>
<td>9.0953</td>
<td>X X X</td>
</tr>
</tbody>
</table>

Regression Analysis: Lines Changed versus CTM, NLM, WMC, CTA, LCOM

The regression equation is

\[ \text{Lines Changed} = 20.3 + 3.86 \text{CTM} - 1.53 \text{NLM} - 15.8 \text{CTA} - 0.0261 \text{LCOM} \]

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Coef</th>
<th>SE Coef</th>
<th>T</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>20.3</td>
<td>7.192</td>
<td>2.82</td>
<td>0.067</td>
</tr>
<tr>
<td>CTM</td>
<td>3.861</td>
<td>1.218</td>
<td>3.17</td>
<td>0.051</td>
</tr>
<tr>
<td>NLM</td>
<td>-1.5269</td>
<td>0.6910</td>
<td>-2.21</td>
<td>0.114</td>
</tr>
<tr>
<td>CTA</td>
<td>-15.796</td>
<td>6.474</td>
<td>-2.44</td>
<td>0.093</td>
</tr>
<tr>
<td>LCOM</td>
<td>-0.0261</td>
<td>0.01254</td>
<td>-2.08</td>
<td>0.129</td>
</tr>
</tbody>
</table>

S = 7.136
R-Sq = 81.3%
R-Sq(adj) = 56.4%

Analysis of Variance

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Seq SS</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>4</td>
<td>664.74</td>
<td>166.18</td>
<td>3.26</td>
<td>0.179</td>
<td></td>
</tr>
<tr>
<td>Residual Error</td>
<td>3</td>
<td>152.76</td>
<td>50.92</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>7</td>
<td>817.50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Regression Analysis: Lines Changed versus CTM, NLM, CTA, LCOM

The regression equation is

\[ \text{Lines Changed} = 110.48 + 11.75 \text{CTM} - 15.95 \text{NLM} - 16.02 \text{CTA} - 0.0261 \text{LCOM} \]

S = 7.136
R-Sq = 81.3%
R-Sq(adj) = 56.4%

Analysis of Variance

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Seq SS</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>4</td>
<td>664.74</td>
<td>166.18</td>
<td>3.26</td>
<td>0.179</td>
<td></td>
</tr>
<tr>
<td>Residual Error</td>
<td>3</td>
<td>152.76</td>
<td>50.92</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>7</td>
<td>817.50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>